Supplementary Data

1 Example images

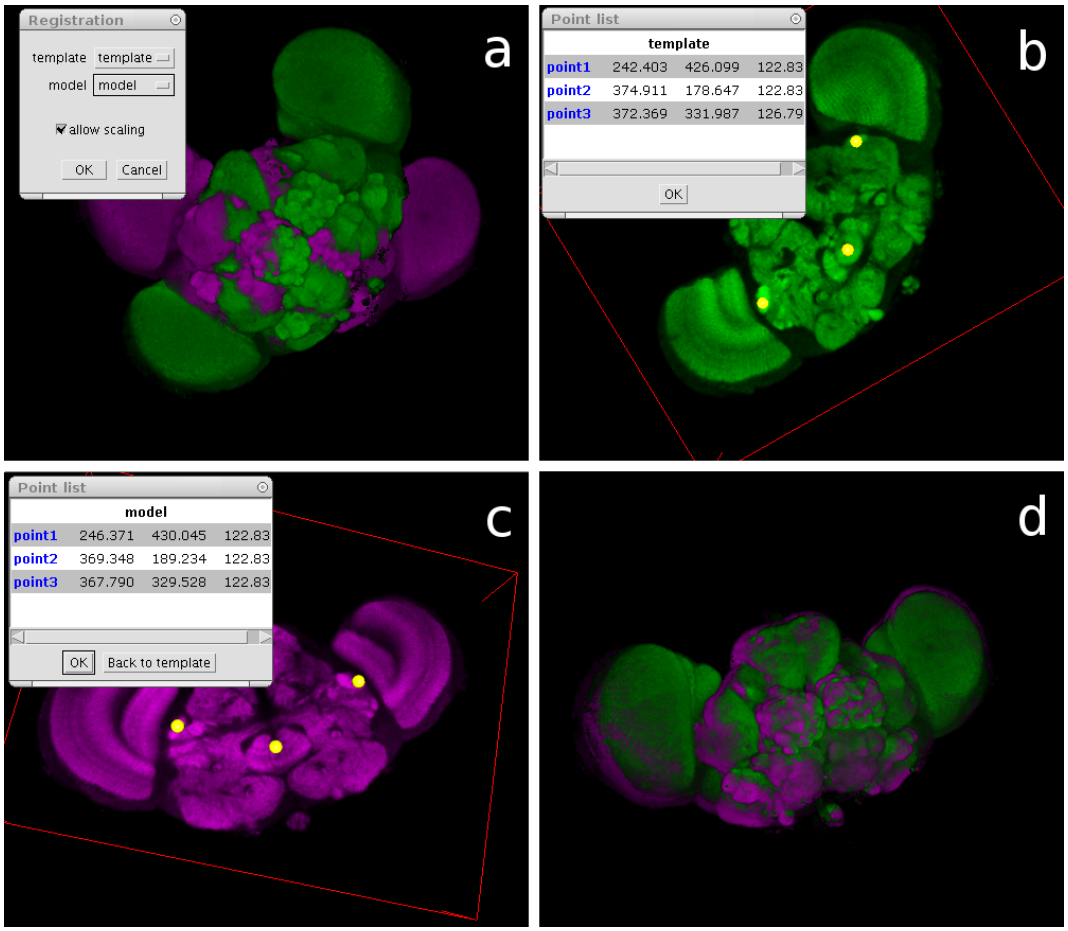

Figure 1: Registration. (a) Two unaligned *Drosophila* brains. The reference brain is shown in green, the model brain (the one to be registered) is shown in magenta. (b) Landmark selection in the model brain. The display mode is automatically changed to orthoslices. The user can scroll through the image planes and select landmarks, which are displayed in a separate dialog window. Landmarks are shown yellow in the figure. They can be moved, highlighted and renamed. (c) Subsequent landmark selection of the template brain. (d) The landmarks are used to infer a rigid transformation which aligns the model image to the landmark image.

2 A more advanced API example simulating dendritic growth in the thorax of a *Drosophila*

We designed our framework with a dual purpose. On the one hand, to provide an interactive 3D scene for end-users. On the other hand, to provide an easy programming interface to instantiate a 3D scene and manipulate its internals. Programmers can trivially augment applications with 3D visualization capabilities. This section demonstrates the latter aspect. A working example is listed. Similar examples and tutorials, together with source code, are found online at `http://3dviewer.neurofly.de`.

```
1  import ij3d.Image3DUniverse;
2  import ij3d.Content;
3
4  import voltex.VoltexVolume;
5  import voltex.VoltexGroup;
6
7  import ij.IJ;
8  import ij.ImageJ;
9  import ij.ImagePlus;
10
11 import java.io.DataInputStream;
12 import java.io.FileInputStream;
13 import java.io.EOFException;
14
15 public class Editable_Volume {
16
17     public static void main(String[] args) throws Exception {
18         new ImageJ();
19
20         // Create a 3D scene and show it
21         Image3DUniverse univ = new Image3DUniverse(600, 480);
22         univ.show();
23
24         // Open the image and add it as a volume
25         ImagePlus image = IJ.openImage("thorax.tif");
26         Content c = univ.addVoltex(image);
27         c.setTransparency(0.7f);
28
29         // Adjust the view
30         univ.rotateToNegativeXY();
31         univ.waitForNextFrame();
32         univ.getViewPlatformTransformer().zoom(40);
33
34         // Retrieve the VoltexVolume of this volume rendering
35         VoltexVolume volume = ((VoltexGroup)c.getContent()).
36                     getRenderer().getVolume();
37
38         // Read in data from file
39         DataInputStream is = new DataInputStream(
40                     new FileInputStream("thorax.stream"));
41
42         while (true) {
43             try {
44                 int x = is.readInt();
45                 int y = is.readInt();
46                 int z = is.readInt();
47                 int v = is.readInt();
48                 volume.set(x, y, z, v);
49             } catch(EOFException ex) {
50                 break;
51             }
52         }
53         is.close();
54     }
55 }
```

Listing 1: The source code here opens an image, renders it as a volume rendering in a virtual universe and subsequently changes voxel values, as read from a stream.

Listing 1 shows how an instance of `Image3DUniverse` is created. An image is opened and added to the universe as a volume rendering via `addVoltex()`. This method returns the new `Content`, which is then used to make it partially transparent.

After adjusting the view, the `VoltexVolume` of the volume rendering is retrieved. The `VoltexVolume` allows to change voxel values in volume renderings directly. The example shows how the values are read from a file input stream, which are interpreted as tuples of 4

consecutive integer values, specifying the x, y and z coordinate and the new voxel value.

The example uses data obtained via a segmentation algorithm. The red channel shows an antibody staining, visualizing the expression pattern of a *Drosophila* GAL4 driver line. This channel was used to simulate axon growth.

The full source code can be found at `http://3dviewer.neurofly.de/?page=bioinf09/source&category=Publications`. Next to the source code, it explains the requirements, how to compile and how to run the example. Additionally, it provides a Java webstart application which offers to run the example with one click (i.e. without requiring installation), and a demo movie.